(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2023년01월31일
(11) 등록번호 10-2494565
(24) 등록일자 2023년01월27일

(51) 국제특허분류(Int. Cl.)
G06N 3/063 (2023.01) *G06N 3/04* (2023.01)
(52) CPC특허분류
G06N 3/063 (2013.01)
G06N 3/04 (2023.01)
(21) 출원번호 10-2020-0060185
(22) 출원일자 2020년05월20일
심사청구일자 2020년05월20일
(65) 공개번호 10-2021-0143440
(43) 공개일자 2021년11월29일
(56) 선행기술조사문헌
KR1020180073118 A*
*는 심사관에 의하여 인용된 문헌

(73) 특허권자
연세대학교 산학협력단
서울특별시 서대문구 연세로 50 (신촌동, 연세대학교)
(72) 발명자
정성욱
서울특별시 서대문구 연세로 50, 연세대학교 제3공학관 C513(신촌동)
이수민
서울특별시 서대문구 연세로 50, 연세대학교 제3공학관 C421(신촌동)
주성환
서울특별시 서대문구 연세로 50, 연세대학교 제3공학관 C421(신촌동)
(74) 대리인
민영준

전체 청구항 수 : 총 14 항 심사관 : 이준상

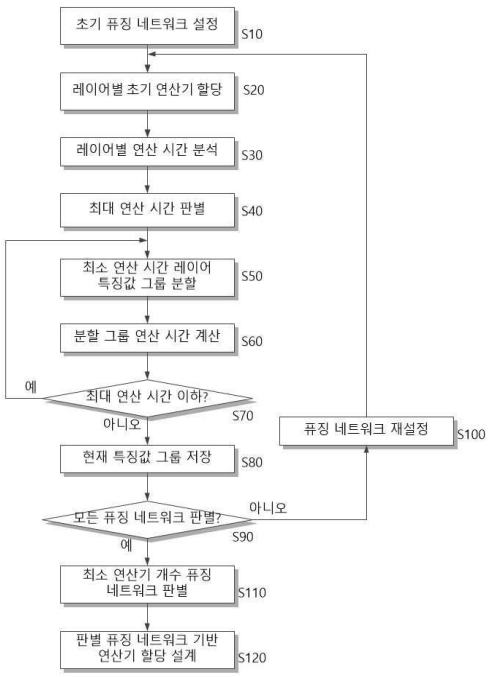
(54) 발명의 명칭 **콘볼루션 신경망의 하드웨어 구조 최적화 방법**

(57) 요 약

본 발명은 다수의 콘볼루션 레이어를 포함하는 콘볼루션 신경망(이하 CNN)의 다수의 콘볼루션 레이어 중 최종단에 위치하는 최종 콘볼루션 레이어가 시스템에서 요구하는 시스템 요구 처리량을 만족하도록 출력할 수 있는 초기 특징값 그룹을 탐색하고, 다른 콘볼루션 레이어에서 출력되는 이전 특징맵들에서 탐색된 상기 초기 특징값 그

(뒷면에 계속)

대 표 도 - 도4



룹에 대응하는 크기의 특징값 그룹을 분석하여 초기 퓨징 네트워크를 설정하는 단계, 상기 초기 퓨징 네트워크 또는 재설정된 퓨징 네트워크를 기반으로 상기 다수의 콘볼루션 레이어 각각에 특징값 그룹의 크기에 대응하는 개수의 연산기를 할당하여 각 레이어별 연산 시간을 계산하고, 계산된 레이어별 연산 시간 중 최소 연산 시간을 갖는 레이어를 판별하여 특징값 그룹의 크기를 기지정된 방식으로 분할하는 단계 및 최종 콘볼루션 레이어의 특징값 그룹의 크기를 기지정된 방식으로 변경하여 퓨징 네트워크를 반복적으로 재설정하고, 각 퓨징 네트워크에서 분할된 특징값 그룹의 크기를 기반으로 할당되는 연산기 개수가 최소인 특징값 그룹의 크기를 판별하여, 상기 다수의 콘볼루션 레이어 각각에 포함될 연산기 개수를 최종 결정하는 단계를 포함하여 요구되는 처리량을 만족할 수 있으면서 연산기 개수를 최소화할 수 있는 콘볼루션 신경망의 하드웨어 구조 최적화 방법을 제공할 수 있다.

명 세 서

청구범위

청구항 1

다수의 콘볼루션 레이어를 포함하는 콘볼루션 신경망(이하 CNN)의 하드웨어 구조 최적화 방법에 있어서,

상기 CNN의 다수의 콘볼루션 레이어 중 최종단에 위치하는 최종 콘볼루션 레이어가 시스템에서 요구하는 시스템 요구 처리량을 만족하도록 출력할 수 있는 초기 특징값 그룹의 크기를 탐색하고, 상기 초기 특징값 그룹의 크기를 기반으로 이전 콘볼루션 레이어들어서 획득해야 하는 특징값 그룹의 크기를 판별하여 초기 퓨징 네트워크를 설정하는 단계;

상기 초기 퓨징 네트워크 또는 재설정된 퓨징 네트워크를 기반으로 상기 다수의 콘볼루션 레이어 각각에 특징값 그룹의 크기에 대응하는 개수의 연산기를 할당하여 각 레이어별 연산 시간을 계산하고, 계산된 레이어별 연산 시간 중 최소 연산 시간을 갖는 레이어를 판별하여 특징값 그룹의 크기를 기지정된 방식으로 분할하는 단계; 및

최종 콘볼루션 레이어의 특징값 그룹의 크기를 기지정된 방식으로 변경하여 퓨징 네트워크를 반복적으로 재설정하고, 각 퓨징 네트워크에서 분할된 특징값 그룹의 크기를 기반으로 할당되는 연산기 개수가 최소인 특징값 그룹의 크기를 판별하여, 상기 다수의 콘볼루션 레이어 각각에 포함될 연산기 개수를 최종 결정하는 단계를 포함하되,

상기 초기 퓨징 네트워크를 설정하는 단계는

상기 시스템 요구 처리량을 판별하는 단계;

상기 최종 콘볼루션 레이어에서 출력되는 최종 특징맵에서 기지정된 축 방향 크기의 약수를 추출하고, 추출된 약수에서 최소 약수로부터 점차 큰 약수를 순차적으로 선택하고, 현재 선택된 약수와 나머지 축 방향의 기지정된 크기에 따라 설정되는 초기 연산 크기를 설정하는 단계;

상기 연산 크기에 따라 상기 최종 콘볼루션 레이어에 요구되는 연산 시간을 판별하는 단계;

판별된 연산 시간 중 상기 시스템 요구 처리량을 만족하는 연산 시간을 갖는 최소 연산 크기를 초기 특징값 그룹으로 설정하는 단계를 포함하는 CNN의 하드웨어 구조 최적화 방법.

청구항 2

삭제

청구항 3

제1항에 있어서, 상기 시스템 요구 처리량을 판별하는 단계는

상기 시스템 요구 처리량을 시스템의 정상 동작을 위해 상기 최종 콘볼루션 레이어가 출력하는 최종 특징맵의 특징값인 최종 특징값을 획득해야 하는 시간인 최종 특징값 획득 요구 시간으로 계산하는 CNN의 하드웨어 구조 최적화 방법.

청구항 4

제3항에 있어서, 상기 연산 시간을 판별하는 단계는

상기 연산 시간을 상기 최종 콘볼루션 레이어에 설정된 연산 크기에 대응하는 개수의 연산기가 할당되는 경우, 상기 최종 콘볼루션 레이어가 상기 최종 특징맵에서 하나의 특징값을 획득하는데 소요되는 최종 특징값 획득 시간으로 계산하는 CNN의 하드웨어 구조 최적화 방법.

청구항 5

제4항에 있어서, 상기 연산 크기를 설정하는 단계는

상기 초기 특징값 그룹으로 설정하는 단계에서 계산된 상기 최종 특징값 획득 시간이 상기 최종 특징값 획득 요구 시간보다 큰 것으로 판별되면, 상기 최종 특징맵에서 기지정된 축 방향 크기의 약수 중 현재 선택된 약수보다 큰 약수를 선택하는 CNN의 하드웨어 구조 최적화 방법.

청구항 6

제5항에 있어서, 상기 초기 특징값 그룹으로 설정하는 단계는

계산된 상기 최종 특징값 획득 시간이 상기 최종 특징값 획득 요구 시간 이하인 것으로 판별되면, 현재 선택된 약수와 나머지 축 방향에서 기지정된 크기의 연산 크기를 초기 특징값 그룹으로 설정하는 CNN의 하드웨어 구조 최적화 방법.

청구항 7

제6항에 있어서, 상기 초기 퓨징 네트워크로 설정하는 단계는

상기 최종 콘볼루션 레이어를 제외한 나머지 콘볼루션 레이어의 적어도 하나의 커널 크기를 기반으로, 상기 나머지 콘볼루션 레이어에서 출력되는 특징맵 각각에서 설정된 초기 특징값 그룹에 대응하는 크기의 특징값 그룹을 판별하여 상기 초기 퓨징 네트워크를 설정하는 단계를 포함하는 CNN의 하드웨어 구조 최적화 방법.

청구항 8

제7항에 있어서, 상기 분할하는 단계는

상기 초기 퓨징 네트워크 또는 재설정된 퓨징 네트워크에 대응하여 다수의 콘볼루션 레이어 각각에 설정된 특징값 그룹의 크기에 대응하는 개수의 연산기를 할당하는 단계;

할당된 연산기 개수에 따라 상기 다수의 콘볼루션 레이어 각각이 대응하는 특징값 그룹을 연산하기 위해 소요하는 연산 시간을 계산하는 단계;

상기 다수의 콘볼루션 레이어 각각에 대해 계산된 연산 시간 중 최대 연산 시간과 최소 연산 시간을 판별하는 단계;

최소 연산 시간을 갖는 콘볼루션 레이어를 판별하고, 판별된 콘볼루션 레이어에 대응하는 특징값 그룹의 크기를 분할하는 단계;

분할된 특징값 그룹의 크기에 따라 연산기의 개수를 저감하여 재할당하여 연산 시간을 재계산하는 단계;

재계산된 연산 시간이 상기 최대 연산 시간 이하이면, 분할된 특징값 그룹의 크기를 저장하고, 재계산된 연산 시간을 기반으로 다시 최소 연산 시간을 갖는 콘볼루션 레이어를 재 판별하는 단계; 및

재계산된 연산 시간이 상기 최대 연산 시간을 초과하면, 이전 마지막 저장된 특징값 그룹의 크기를 현재 설정된 퓨징 네트워크의 최적 특징값 그룹 크기로 판별하는 단계를 포함하는 CNN의 하드웨어 구조 최적화 방법.

청구항 9

제8항에 있어서, 상기 특징값 그룹의 크기를 분할하는 단계는

상기 특징값 그룹의 각 축 방향 크기를 판별하는 단계;

판별된 축 방향 크기 각각의 약수의 개수를 분석하는 단계;

분석된 축 방향 약수의 개수 중 최대 개수의 약수를 갖는 축 방향을 선택하는 단계; 및

상기 특징값 그룹을 선택된 축 방향으로 분할하는 단계를 포함하는 CNN의 하드웨어 구조 최적화 방법.

청구항 10

제9항에 있어서, 상기 축 방향으로 분할하는 단계는

선택된 축 방향 약수 중 1을 제외한 최소 약수로 상기 특징값 그룹을 분할하는 CNN의 하드웨어 구조 최적화 방법.

청구항 11

제8항에 있어서, 상기 최종 결정하는 단계는

상기 최종 특징맵에서 기지정된 축 방향 크기에서 추출된 약수 중 초기 특징값 그룹으로 설정된 약수보다 큰 약수가 선택되어 설정되는 특징값 그룹에 대응하여 다수의 퓨징 네트워크를 반복 재설정하는 단계;

반복 재설정되는 다수의 퓨징 네트워크 각각에 대해 최소 연산 시간을 갖는 콘볼루션 레이어에 대응하는 특징값 그룹의 크기를 분할하여 판별된 최적 특징값 그룹 크기에 따라 요구되는 전체 연산기 개수를 비교하는 단계; 및

요구되는 연산기 개수가 최소인 최적 특징값 그룹의 크기를 판별하고, 판별된 최적 특징값 그룹의 크기에 대응하는 퓨징 네트워크에 따라 상기 다수의 콘볼루션 레이어 각각에 포함될 연산기 개수를 결정하는 단계를 포함하는 CNN의 하드웨어 구조 최적화 방법.

청구항 12

제11항에 있어서, 상기 CNN은

파이프 라인 기법에 따라 다수의 콘볼루션 레이어가 연산을 수행하는 CNN의 하드웨어 구조 최적화 방법.

청구항 13

제11항에 있어서, 상기 다수의 콘볼루션 레이어 각각은

이전 콘볼루션 레이어에서 상기 퓨징 네트워크에 의해 지정되는 특징값 그룹에 대한 연산이 수행되면, 연산된 특징값 그룹을 인가받아 연산을 수행하는 CNN의 하드웨어 구조 최적화 방법.

청구항 14

제11항에 있어서, 상기 CNN은

FHD 이미지를 UHD 이미지로 업 스케일링하는 FSRCNN-s을 하드웨어로 구현되는 CNN의 하드웨어 구조 최적화 방법.

청구항 15

제1항 및 제3항 내지 제14항 중 어느 한 항의 CNN의 하드웨어 구조 최적화 방법을 구현하기 위한 프로그램 명령어가 기록된 기록매체.

발명의 설명

기 술 분 야

[0001] 본 발명은 콘볼루션 신경망의 하드웨어 구조 최적화 방법에 관한 것으로, 콘볼루션 신경망의 레이어 간 지연시간 차이 최소화를 통한 하드웨어 구조 최적화 방법에 관한 것이다.

배 경 기 술

[0002] 인공 신경망은 대량의 학습 데이터를 이용한 딥 러닝(Deep Learning) 방식으로 학습되어 기존의 알고리즘보다 매우 우수한 성능을 나타낸다. 이로 인해 최근 다양한 분야에서 인공 신경망이 적용되고 있다.

[0003] 그리고 인공 신경망을 이용하는 분야 중 하나로 낮은 해상도의 이미지를 더 높은 해상도의 이미지로 변환하는 이미지 업 스케일링(Image Up scaling) 기술이 관심을 받고 있다. 최근 이미지 업 스케일링 기술은 일 예로 FHD(Full High Definition) 이미지를 UHD(Ultra High Definition) 이미지로 변환하기 위해 적용되고 있다. 이와 같은 이미지 업 스케일링 기술을 적용하면, FHD 이미지를 그대로 전송하여, 사용자 단말에서 UHD의 높은 해상도 이미지로 변환할 수 있으므로, 기존 제작된 다수의 FHD 콘텐츠를 효과적으로 재사용할 수 있을 뿐만 아니라 적은 통신 대역을 사용한다는 장점이 있다.

[0004] 인공 신경망을 이용하여 수행되는 이미지 업 스케일링 기법은 슈퍼 레졸루션(Super-Resolution) 기법으로도 잘 알려져 있으며, 슈퍼 레졸루션 기법에서는 주로 이미지 처리 분야에서 매우 우수한 성능을 나타내는 인공 신경망인 콘볼루션 신경망(Convolutional Neural Network: 이하 CNN)이 이용되고 있다. 그러나 CNN과 같은 인공

신경망은 높은 계산 복잡도로 인해, 이를 하드웨어로 구현이 어렵다는 한계가 있다.

[0005] 일 예로 슈퍼 레졸루션 기법에 적용되는 CNN 알고리즘인 FSRCNN-s(mall model size version of Fast Super-Resolution Convolutional Neural Network)을 하드웨어로 구현하는 경우, 1920×1080의 해상도를 갖는 FHD 이미지를 3840×2160 해상도를 갖는 UHD 이미지로 업 스케일링 하기위해서는 5.2GOP의 연산을 수행해야 한다. 따라서 초당 60프레임의 FHD 이미지를 실시간으로 업 스케일링 하기위해서는 311.7GOP 수준의 처리량(throughput)이 요구된다. 따라서 요구되는 처리량을 만족하기 위해서는 대량의 연산기를 이용한 병렬 연산 처리가 수행되어야 한다.

[0006] 그러나 이러한 대량의 연산기를 하드웨어로 구현하기는 어려우며, 특히 단일 칩 셋의 하드웨어에 집적하여 구현하기는 현실적으로 불가능한다. 그러므로 요구되는 처리량으로 동작하는 인공 신경망의 하드웨어 구조를 최적화할 필요가 있다.

선행기술문헌

특허문헌

[0007] (특허문헌 0001) 한국 공개 특허 제10-2019-0087265호 (2019.07.24 공개)

발명의 내용

해결하려는 과제

[0008] 본 발명의 목적은 콘볼루션 신경망에 요구되는 처리량을 만족하면서 연산기 개수를 최소화하여 하드웨어 구조 최적화할 수 있는 콘볼루션 신경망의 하드웨어 구조 최적화 방법을 제공하는데 있다.

[0009] 본 발명의 다른 목적은 요구되는 처리량을 기준으로 콘볼루션 신경망의 다수의 레이어 중 최종 레이어에서 동시에 병렬 연산이 수행되어야 하는 최소 크기를 설정하고, 설정된 최소 크기에 따라 이전 레이어가 한번에 수행할 연산 크기를 조절하여, 병목 현상이 발생되지 않고 파이프 라인 구조로 연산이 수행되도록 할 수 있는 콘볼루션 신경망의 하드웨어 구조 최적화 방법을 제공하는데 있다.

과제의 해결 수단

[0010] 상기 목적을 달성하기 위한 본 발명의 일 실시예에 따른 콘볼루션 신경망의 하드웨어 구조 최적화 방법은 다수의 콘볼루션 레이어를 포함하는 콘볼루션 신경망(이하 CNN)의 다수의 콘볼루션 레이어 중 최종단에 위치하는 최종 콘볼루션 레이어가 시스템에서 요구하는 시스템 요구 처리량을 만족하도록 출력할 수 있는 초기 특징값 그룹을 탐색하고, 다른 콘볼루션 레이어에서 출력되는 이전 특징맵들에서 탐색된 상기 초기 특징값 그룹에 대응하는 크기의 특징값 그룹을 분석하여 초기 퓨징 네트워크를 설정하는 단계; 상기 초기 퓨징 네트워크 또는 재설정된 퓨징 네트워크를 기반으로 상기 다수의 콘볼루션 레이어 각각에 특징값 그룹의 크기에 대응하는 개수의 연산기를 할당하여 각 레이어별 연산 시간을 계산하고, 계산된 레이어별 연산 시간 중 최소 연산 시간을 갖는 레이어를 판별하여 특징값 그룹의 크기를 기지정된 방식으로 분할하는 단계; 및 최종 콘볼루션 레이어의 특징값 그룹의 크기를 기지정된 방식으로 변경하여 퓨징 네트워크를 반복적으로 재설정하고, 각 퓨징 네트워크에서 분할된 특징값 그룹의 크기를 기반으로 할당되는 연산기 개수가 최소인 특징값 그룹의 크기를 판별하여, 상기 다수의 콘볼루션 레이어 각각에 포함될 연산기 개수를 최종 결정하는 단계를 포함한다.

[0011] 상기 초기 퓨징 네트워크를 설정하는 단계는 상기 시스템 요구 처리량을 판별하는 단계; 상기 최종 콘볼루션 레이어에서 출력되는 최종 특징맵에서 기지정된 축 방향 크기의 약수를 추출하고, 추출된 약수에서 최소 약수로부터 점차 큰 약수를 순차적으로 선택하고, 현재 선택된 약수와 나머지 축 방향의 기지정된 기에 따라 설정되는 초기 연산 크기를 설정하는 단계; 상기 연산 크기에 따라 상기 최종 콘볼루션 레이어에 요구되는 연산 시간을 판별하는 단계; 판별된 연산 시간 중 상기 시스템 요구 처리량을 만족하는 연산 시간을 갖는 최소 연산 크기를 초기 특징값 그룹으로 설정하는 단계; 및 상기 최종 콘볼루션 레이어를 제외한 나머지 콘볼루션 레이어 각각으로부터 출력되는 특징맵 각각에서 상기 초기 특징값 그룹에 대응하는 크기의 특징값 그룹을 판별하여 상기 초기 퓨징 네트워크로 설정하는 단계를 포함할 수 있다.

[0012] 상기 시스템 요구 처리량을 판별하는 단계는 상기 시스템 요구 처리량을 시스템의 정상 동작을 위해 상기 최종

콘볼루션 레이어가 출력하는 최종 특징맵의 특징값인 최종 특징값을 획득해야 하는 시간인 최종 특징값 획득 요구 시간으로 계산할 수 있다.

[0013] 상기 연산 시간을 판별하는 단계는 상기 연산 시간을 상기 최종 콘볼루션 레이어에 설정된 연산 크기에 대응하는 개수의 연산기가 할당되는 경우, 상기 최종 콘볼루션 레이어가 상기 최종 특징맵에서 하나의 특징값을 획득하는데 소요되는 최종 특징값 획득 시간으로 계산할 수 있다.

[0014] 상기 연산 크기를 설정하는 단계는 상기 초기 특징값 그룹으로 설정하는 단계에서 계산된 상기 최종 특징값 획득 시간이 상기 최종 특징값 획득 요구 시간보다 큰 것으로 판별되면, 상기 최종 특징맵에서 기지정된 축 방향 크기의 약수 중 현재 선택된 약수보다 큰 약수를 선택할 수 있다.

[0015] 상기 초기 특징값 그룹으로 설정하는 단계는 계산된 상기 최종 특징값 획득 시간이 상기 최종 특징값 획득 요구 시간 이하인 것으로 판별되면, 현재 선택된 약수와 나머지 축 방향에서 기지정된 크기의 연산 크기를 초기 특징값 그룹으로 설정할 수 있다.

[0016] 상기 초기 퓨징 네트워크로 설정하는 단계는상기 최종 콘볼루션 레이어를 제외한 나머지 콘볼루션 레이어의 적어도 하나의 커널 크기를 기반으로, 상기 나머지 콘볼루션 레이어에서 출력되는 특징맵 각각에서 설정된 초기 특징값 그룹에 대응하는 크기의 특징값 그룹을 판별하여 상기 초기 퓨징 네트워크를 설정하는 단계를 포함할 수 있다.

[0017] 상기 분할하는 단계는 상기 초기 퓨징 네트워크 또는 재설정된 퓨징 네트워크에 대응하여 다수의 콘볼루션 레이어 각각에 설정된 특징값 그룹의 크기에 대응하는 개수의 연산기를 할당하는 단계; 할당된 연산기 개수에 따라 상기 다수의 콘볼루션 레이어 각각이 대응하는 특징값 그룹을 연산하기 위해 소요하는 연산 시간을 계산하는 단계; 상기 다수의 콘볼루션 레이어 각각에 대해 계산된 연산 시간 중 최대 연산 시간과 최소 연산 시간을 판별하는 단계; 최소 연산 시간을 갖는 콘볼루션 레이어를 판별하고, 판별된 콘볼루션 레이어에 대응하는 특징값 그룹의 크기를 분할하는 단계; 분할된 특징값 그룹의 크기에 따라 연산기의 개수를 저감하여 재할당하여 연산 시간을 재계산하는 단계; 재계산된 연산 시간이 상기 최대 연산 시간 이하이면, 분할된 특징값 그룹의 크기를 저장하고, 재계산된 연산 시간을 기반으로 다시 최소 연산 시간을 갖는 콘볼루션 레이어를 재 판별하는 단계; 및 재계산된 연산 시간이 상기 최대 연산 시간을 초과하면, 이전 마지막 저장된 특징값 그룹의 크기를 현재 설정된 퓨징 네트워크의 최적 특징값 그룹 크기로 판별하는 단계를 포함할 수 있다.

[0018] 상기 특징값 그룹의 크기를 분할하는 단계는 상기 특징값 그룹의 각 축 방향 크기를 판별하는 단계; 판별된 축 방향 크기 각각의 약수의 개수를 분석하는 단계; 분석된 축 방향 약수의 개수 중 최대 개수의 약수를 갖는 축 방향을 선택하는 단계; 및 상기 특징값 그룹을 선택된 축 방향으로 분할하는 단계를 포함할 수 있다.

[0019] 상기 축 방향으로 분할하는 단계는 선택된 축 방향 약수 중 1을 제외한 최소 약수로 상기 특징값 그룹을 분할할 수 있다.

[0020] 상기 최종 결정하는 단계는 상기 최종 특징맵에서 기지정된 축 방향 크기에서 추출된 약수 중 초기 특징값 그룹으로 설정된 약수보다 큰 약수가 선택되어 설정되는 특징값 그룹에 대응하여 다수의 퓨징 네트워크를 반복 재설정하는 단계; 반복 재설정되는 다수의 퓨징 네트워크 각각에 대해 최소 연산 시간을 갖는 콘볼루션 레이어에 대응하는 특징값 그룹의 크기를 분할하여 판별된 최적 특징값 그룹 크기에 따라 요구되는 전체 연산기 개수를 비교하는 단계; 및 요구되는 연산기 개수가 최소인 최적 특징값 그룹의 크기를 판별하고, 판별된 최적 특징값 그룹의 크기에 대응하는 퓨징 네트워크에 따라 상기 다수의 콘볼루션 레이어 각각에 포함될 연산기 개수를 결정하는 단계를 포함할 수 있다.

[0021] 상기 CNN은 파이프 라인 기법에 따라 다수의 콘볼루션 레이어가 연산을 수행할 수 있다.

[0022] 상기 다수의 콘볼루션 레이어 각각은 이전 콘볼루션 레이어에서 상기 퓨징 네트워크에 의해 지정되는 특징값 그룹에 대한 연산이 수행되면, 연산된 특징값 그룹을 인가받아 연산을 수행할 수 있다.

[0023] 상기 CNN은 FHD 이미지를 UHD 이미지로 업 스케일링하는 FSRCNN-s을 하드웨어로 구현될 수 있다.

발명의 효과

[0024] 따라서, 본 발명의 실시예에 따른 콘볼루션 신경망의 하드웨어 구조 최적화 방법은 콘볼루션 신경망의 다수의 레이어 중 최종 레이어에서 수행되는 최소 연산 크기를 기준으로 역방향 순서로 이전 레이어가 한번에 수행할 연산 크기를 단계적으로 조절하여, 병목 현상이 발생되지 않고 파이프 라인 구조로 연산이 수행되도록

함으로써, 요구되는 처리량을 만족할 수 있으면서 연산기 개수를 최소화할 수 있다.

도면의 간단한 설명

[0025] 도 1은 FSRCNN_s의 구조를 나타낸다.

도 2는 콘볼루션 연산 방식에 따라 콘볼루션 레이어가 콘볼루션 연산을 시작하기 위해 이전 콘볼루션 레이어에서 미리 계산되어야 하는 데이터 크기를 설명하기 위한 도면이다.

도 3은 도 1의 FSRCNN_s를 파이프 라인 기법으로 동작시키는 경우, 각 레이어별 대기 시간을 나타낸다.

도 4는 본 발명의 일 실시예에 따른 CNN의 하드웨어 구조 최적화 방법을 나타낸다.

도 5는 도 4의 초기 퓨징 네트워크 설정 단계를 상세하게 나타낸 도면이다.

발명을 실시하기 위한 구체적인 내용

[0026] 본 발명과 본 발명의 동작상의 이점 및 본 발명의 실시에 의하여 달성되는 목적을 충분히 이해하기 위해서는 본 발명의 바람직한 실시예를 예시하는 첨부 도면 및 첨부 도면에 기재된 내용을 참조하여야만 한다.

[0027] 이하, 첨부한 도면을 참조하여 본 발명의 바람직한 실시예를 설명함으로써, 본 발명을 상세히 설명한다. 그러나, 본 발명은 여러 가지 상이한 형태로 구현될 수 있으며, 설명하는 실시예에 한정되는 것이 아니다. 그리고, 본 발명을 명확하게 설명하기 위하여 설명과 관계없는 부분은 생략되며, 도면의 동일한 참조부호는 동일한 부재임을 나타낸다.

[0028] 명세서 전체에서, 어떤 부분이 어떤 구성요소를 "포함"한다고 할 때, 이는 특별히 반대되는 기재가 없는 한 다른 구성요소를 제외하는 것이 아니라, 다른 구성요소를 더 포함할 수 있는 것을 의미한다. 또한, 명세서에 기재된 "...부", "...기", "모듈", "블록" 등의 용어는 적어도 하나의 기능이나 동작을 처리하는 단위를 의미하며, 이는 하드웨어나 소프트웨어 또는 하드웨어 및 소프트웨어의 결합으로 구현될 수 있다.

[0029] 도 1은 FSRCNN_s의 구조를 나타내고, 도 2는 콘볼루션 연산 방식에 따라 콘볼루션 레이어가 콘볼루션 연산을 시작하기 위해 이전 콘볼루션 레이어에서 미리 계산되어야 하는 데이터 크기를 설명하기 위한 도면이며, 도 3은 도 1의 FSRCNN_s를 파이프 라인 기법으로 동작시키는 경우, 각 레이어별 대기 시간을 나타낸다.

[0030] 도 1에서도 슈퍼 레졸루션 기법의 대표적인 일 예로 FSRCNN-s의 구조를 도시하였으며, 1920×1080의 해상도를 갖는 FHD 이미지 입력 이미지로 인가받아 3840×2160 해상도를 갖는 UHD 이미지로 업 스케일링하여 출력 이미지로 출력하는 FSRCNN-s의 구조를 도시하였다.

[0031] 도 1을 참조하면, FSRCNN-s는 5개의 콘볼루션 레이어(L1 ~ L5)를 포함하여 구성될 수 있다. 5개의 콘볼루션 레이어(L1 ~ L5) 각각은 기지정된 크기의 적어도 하나의 커널을 포함하여 구성된다. 여기서 적어도 하나의 커널 각각은 학습을 통해 획득되는 다수의 가중치가 커널 크기에 대응하여 배열된 구조를 갖는다.

[0032] FSRCNN_s에서 5개의 콘볼루션 레이어(L1 ~ L5) 중 제1 콘볼루션 레이어(L1)는 폭(Width)×높이(Height)×깊이(Depth)로 표현하여 5×5×1 크기를 갖는 32개의 커널(5×5×1, 32)을 포함하고, 제2 콘볼루션 레이어(L2)는 1×1×32 크기를 갖는 5개의 커널(1×1×32, 5)을 포함하며, 제3 콘볼루션 레이어(L3)는 3×3×5 크기를 갖는 5개의 커널(3×3×5, 5)을 포함한다. 그리고 제4 콘볼루션 레이어(L4)는 1×1×5 크기를 갖는 32개의 커널(1×1×5, 32)을 포함하고, 제5 콘볼루션 레이어(L5)는 3×3×32 크기를 갖는 4개의 커널(3×3×32, 4)을 포함한다.

[0033] 5개의 콘볼루션 레이어(L1 ~ L5) 중 제1 콘볼루션 레이어(L1)는 입력 이미지(IN)를 인가받고, 인가된 입력 이미지에 대해 포함된 커널을 이용하여 특징을 추출함으로써, 제1 특징맵(FM1)을 획득한다. 그리고 제2 내지 제5 콘볼루션 레이어(L2 ~ L5)는 각각 이전단의 콘볼루션 레이어(L1 ~ L3)에서 출력되는 제1 내지 제4 특징맵(FM1 ~ FM4)을 인가받고, 인가된 제1 내지 제3 특징맵(FM1 ~ FM4)의 특징을 추출하여 제2 내지 제5 특징맵(FM2 ~ FM5)을 획득한다.

[0034] 여기서 FSRCNN_s가 다수의 콘볼루션 레이어(L1 ~ L5)를 포함하여, 계층적으로 특징을 추출하여 특징맵을 획득하는 것은 입력 이미지로부터 보다 고차원 특징(High-level Feature)을 추출할 수 있도록 하기 위함이다.

[0035] 일반적인 FSRCNN_s에서는 1092×1080×32의 크기를 갖는 제4 특징맵(FM4)으로부터 3840×2160 해상도를 갖는 UHD 이미지를 곧바로 획득하기 위해, 4개의 콘볼루션 레이어(L1 ~ L4)와 하나의 디콘볼루션 레이어(Deconvolution Layer)를 이용하도록 구성되지만, 디콘볼루션 레이어는 대량의 연산을 요구하므로, 최근에는 연

산량을 대폭 저감시킬 수 있는 셔플링 레이어(Shuffling Layer)가 디콘볼루션 레이어를 대체하여 주로 사용되고 있다. 셔플링 레이어의 경우, 이전 4개의 콘볼루션 레이어(L1 ~ L4)와 마찬가지로 콘볼루션 연산을 수행하는 제5 콘볼루션 레이어(L5)로 구현되어 제5 특징맵(FM5)을 획득하고, 제5 특징맵(FM5)에 대해 별도로 리맵핑(Remapping)을 수행하여 UHD 이미지를 획득하도록 구성된다.

[0036] 이에 도 1에서는 FSRCNN_s가 셔플링 레이어를 포함하여 5개의 콘볼루션 레이어를 포함하는 경우를 가정하여 도시하였으며, 후처리 작업인 리맵핑 작업은 제5 특징맵(FM5)의 각 원소를 기지정된 위치에 재배열하는 작업으로 연산기가 요구되지 않는 작업이므로 여기서는 별도로 도시하지 않았다.

[0037] 한편, 도 1에 도시된 바와 같이 다수의 콘볼루션 레이어(L1 ~ L5)로 구성된 FSRCNN_s를 하드웨어로 구현하는 경우, 다양한 하드웨어 구조를 고려할 수 있다. 일 예로 직렬 및 병렬 구조를 고려할 수 있다.

[0038] 병렬 구조로 다수의 콘볼루션 레이어(L1 ~ L5)로 구성된 FSRCNN_s를 하드웨어를 구현하는 경우에는, 다수의 콘볼루션 레이어(L1 ~ L5) 각각에서 동시에 수행되어야 하는 전체 연산 횟수에 대응하는 개수의 연산기를 포함하도록 구성하여, 다수의 콘볼루션 레이어(L1 ~ L5)이 동시에 연산을 수행하도록 할 수 있다. 일 예로 도 1의 FSRCNN_s에서는 입력 이미지(IN) 및 다수의 특징맵(FM1 ~ FM5)과 다수의 콘볼루션 레이어(L1 ~ L5) 각각에 포함되는 커널의 크기 및 개수에 대응하는 개수의 연산기를 포함하도록 하드웨어를 구현할 수 있다.

[0039] 이와 같이, 병렬 구조로 하드웨어를 구현하는 경우에 단일 프레임의 입력 이미지(IN)에 대해서는 콘볼루션 레이어(L1 ~ L5)단위로 순차 연산을 수행되지만, 다수 프레임의 입력 이미지(IN)가 인가되는 경우, 다수의 콘볼루션 레이어(L1 ~ L5) 각각은 서로 다른 프레임의 입력 이미지에 대해 연산을 수행함으로써, 연산 속도를 대폭 향상시킬 수 있다. 즉 파이프 라인 기법을 적용함으로써 시스템이 요구하는 처리량을 만족시킬 수 있다. 그러나 다수의 콘볼루션 레이어(L1 ~ L5)에 포함되는 커널의 개수 및 크기가 서로 상이하므로, 파이프 라인 기법을 적용하더라도 각 콘볼루션 레이어(L1 ~ L5)별 연산 시간에 차이가 발생한다는 문제가 있다.

[0040] 만일 FSRCNN_s의 콘볼루션 레이어(L1 ~ L5) 각각이 입력 이미지(IN) 및 다수의 특징맵(FM1 ~ FM5)과 다수의 콘볼루션 레이어(L1 ~ L5) 각각에 포함되는 커널의 크기 및 개수의 곱에 대응하는 개수의 연산기를 구비한다면, 각 콘볼루션 레이어(L1 ~ L5)가 단일 클럭에 요구되는 연산을 수행할 수 있으므로, 콘볼루션 레이어(L1 ~ L5)별 연산 시간에 차이가 발생하지 않으나, 이와 같은 완전 병렬 구조로 FSRCNN_s를 구현하기 위해서는 대규모의 연산기를 요구하여, 현실적으로 하드웨어로 구현이 불가능하다.

[0041] 반면, 직렬 구조로 다수의 콘볼루션 레이어(L1 ~ L5)로 구성된 FSRCNN_s를 하드웨어를 구현하는 경우, 적어도 하나의 연산기만을 구비하고, 구비된 적어도 하나의 연산기가 FSRCNN_s에서 요구되는 연산 전체를 수행하도록 구성될 수도 있다. 즉 요구되는 연산기 개수를 최소화할 수 있다. 그러나 이 경우, 연산 속도의 저하로 인해 시스템에서 요구하는 처리량을 만족할 수 없다는 한계가 있다.

[0042] 다만 다수의 콘볼루션 레이어(L1 ~ L5) 각각에 적어도 하나의 연산기가 구비되는 경우, 콘볼루션 레이어(L1 ~ L5) 각각은 이전 단에 배치된 콘볼루션 레이어의 일부 연산 결과를 기반으로 먼저 연산을 수행할 수 있다. 즉 이전 단에 배치된 콘볼루션 레이어가 연산을 완료하지 않더라도, 파이프 라인 기법에 따라 먼저 연산을 수행할 수 있다.

[0043] 도 2에서는 일예로 이전 단에 배치된 콘볼루션 레이어가 3×5 크기의 특징맵(FM)을 획득하고, 연산을 수행해야 하는 콘볼루션 레이어의 커널이 3×3 크기를 갖는 경우에, 콘볼루션 연산 기법에 따라 콘볼루션 레이어가 연산을 시작하기 위해 이전 단에 배치된 콘볼루션 레이어가 미리 획득해야 하는 특징값의 크기를 설명하기 위한 도면이다.

[0044] 도 2를 참조하면, 연산을 수행해야 하는 콘볼루션 레이어가 3×3 크기의 커널을 가지므로, 콘볼루션 레이어는 3×5 크기의 특징맵(FM) 중에서 특징값 그룹(g1)에 해당하는 3×3 크기의 특징값만이 획득되면, y_{11}의 특징값을 획득하기 위한 연산을 수행할 수 있다. 즉 이전 단에 배치된 콘볼루션 레이어가 3×5 크기의 특징맵에 포함되는 모든 특징값을 계산하지 않고, 3×3 크기의 특징값 그룹(g1)만을 획득하여도 y_{11}의 특징값을 획득하기 위한 연산을 수행할 수 있다. 유사하게 콘볼루션 레이어는 특징값 그룹(g2)에 해당하는 3×3 크기의 특징값만이 획득되면, y_{12}의 특징값을 획득하기 위한 연산을 시작할 수 있다. 그리고 콘볼루션 레이어가 y_{11}의 특징값과 y_{12}의 특징값을 획득하기 위한 연산을 수행하는 동안에는 이전 콘볼루션 레이어가 특징값 그룹(g3)에 해당하는 특징값들을 미리 획득할지라도 이용되지 않는다.

[0045] 이는 다수의 콘볼루션 레이어(L1 ~ L5) 각각이 이전 단에 배치된 콘볼루션 레이어에서 일부 연산만 수행되더라

도, 획득된 특징값을 기반으로 연산을 시작할 수 있음을 의미할 뿐만 아니라, 현재 연산에 이용되는 특징값 이외의 특징값을 이전 단에 배치된 콘볼루션 레이어가 미리 획득할지라도 이용되지 않을 의미한다.

[0046] 이 때, 다수의 콘볼루션 레이어(L1 ~ L5) 각각이 연산을 시작하기 위해 이전 계산되어야 하는 특징값의 개수, 즉 특징값 그룹의 크기는 각 콘볼루션 레이어(L1 ~ L5)에 포함되는 커널의 크기 및 개수에 따라 결정될 수 있다.

[0047] 도 1에서는 상기한 바와 같이, FSRCNN_s의 다수의 콘볼루션 레이어(L1 ~ L5) 각각이 파이프 라인 기법에 따라 부분 연산을 수행하는 경우, 각 콘볼루션 레이어(L1 ~ L5)에 인가되는 입력 이미지(IN) 또는 다수의 특징맵(FM1 ~ FM5)에서 미리 획득되어야 하는 특징값의 크기를 나타낸다.

[0048] 도 1을 참조하면, 파이프 라인 기법 적용시 최종적으로 연산을 수행하는 것은 제5 콘볼루션 레이어(L5)이고, 제5 콘볼루션 레이어(L5)가 제5 특징맵(FM5)의 a×b×4 크기의 특징값을 추출하기 위해서는 제4 특징맵(FM4)의 대응하는 위치에 (a+2)×(b+2)×32 크기의 특징값이 우선 추출되어 있어야 한다. 유사하게 제4 콘볼루션 레이어(L4)가 제4 특징맵(FM4)의 (a+2)×(b+2)×32 크기의 특징값을 추출하기 위해서는 제3 특징맵(FM3)의 대응하는 위치에 (a+2)×(b+2)×5 크기의 특징값이 우선 추출되어 있어야 한다. 이와 같이 각 콘볼루션 레이어가 연산을 수행하기 위해 이전 미리 획득되어야 하는 크기의 특징값 그룹을 본 실시예에서는 퓨징 네트워크(Fusing network)라고 한다. 즉 도 1은 FSRCNN_s에서 각 콘볼루션 레이어에 인가되는 특징맵의 퓨징 네트워크를 도식화한 것이다.

[0049] 도 1에 도시된 바와 같이, FSRCNN_s의 각 콘볼루션 레이어(L1 ~ L5)가 파이프 라인 기법에 따라 부분 연산을 수행하기 위해서는 이전 콘볼루션 레이어에서 퓨징 네트워크에 대응하는 크기의 연산이 미리 수행되어야만 하며, 이때 각 콘볼루션 레이어(L1 ~ L5)에 요구되는 연산 횟수가 서로 상이하다. 콘볼루션 레이어(L1 ~ L5)별 연산 횟수의 차이는 연산 시간의 차이를 발생시키고, 연산 시간의 차이는 다수의 콘볼루션 레이어(L1 ~ L5) 중 일부 콘볼루션 레이어가 자신이 현재 수행해야하는 모든 연산을 수행하였음에도 이전 콘볼루션 레이어로부터 요구되는 크기의 특징값이 인가되지 않아 연산을 수행하지 못하는 대기 시간(Standby)을 유발하게 된다.

[0050] 표 1은 도 1의 FSRCNN_s의 다수의 콘볼루션 레이어(L1 ~ L5) 각각에 포함되는 커널의 크기(Row×Column×Depth)와 개수(# of Kernel)를 나타낸다.

표 1

[0051]

	L1	L2	L3	L4	L5
Row(K_x)	5	1	3	1	3
Column(K_y)	5	1	3	1	3
Depth(K_z)	1	32	5	5	32
# of Kernel(K_n)	32	5	5	32	4

[0052] 표 1에 도시된 콘볼루션 레이어(L1 ~ L5)별 커널의 크기와 개수 및 콘볼루션 연산의 연산 방식을 고려할 때, 파이프 라인 기법에 따라 제5 콘볼루션 레이어(L5)가 연산을 시작하기 위해서는, 최소한 제4 특징맵(FM4)에서 제5 콘볼루션 레이어(L5)의 커널의 크기에 대응하여 3×3×32 개의 특징값이 미리 계산되어야만 한다. 여기서 제4 콘볼루션 레이어(L4)의 커널의 크기는 1×1×5 크기의 32개의 커널을 구비하여 연산을 수행할 수 있으므로, 제5 콘볼루션 레이어(L5)가 연산을 시작하기 위해서는 제4 콘볼루션 레이어(L4)가 제4 콘볼루션 레이어(L4)의 커널의 크기(1×1×5)와 제5 콘볼루션 레이어(L5)의 커널 크기 중 행 및 열 방향 크기(3×3)를 곱한 횟수(1×1×5×3×3 = 1440) 만큼의 연산을 미리 수행해야만 한다.

[0053] 이는 한번의 연산에 1 클럭(Clock)이 소요된다고 가정하는 경우, 제5 콘볼루션 레이어(L5)가 연산을 시작하기 위해서는 제4 콘볼루션 레이어(L4)에서 1,440 클럭의 연산이 먼저 수행되어야 한다.

[0054] 도 3에서는 이와 같이 다수의 콘볼루션 레이어(L1 ~ L5) 각각이 파이프 라인 기법에 딸 연산을 시작하기 위해, 이전 레이어가 미리 수행할 것이 요구되는 연산 시간을 클럭 단위로 표시하였다.

[0055] 도 3를 살펴보면, 제1 내지 제 5 콘볼루션 레이어(L1 ~ L5) 각각은 서로 다른 커널의 크기와 개수가 서로 상이하여 다음 콘볼루션 레이어가 동작할 수 있도록 미리 연산해야 하는 연산 시간이 서로 상이함을 알 수 있다. 이 중 제2 콘볼루션 레이어(L2)와 제4 콘볼루션 레이어(L4)의 경우, 동일하게 1,440 클럭의 연산이 수행되어야 하므로, 긴 연산 시간을 필요로 한다. 그에 비해 제1, 제3 및 제5 콘볼루션 레이어(L1, L3, L5)의 경우, 800 클럭과 255 클럭 및 1152 클럭의 연산이 수행되어야 한다. 이와 같은 콘볼루션 레이어(L1 ~ L5) 사이의 연산 시간 차이는 FSRCNN_s의 다수의 콘볼루션 레이어(L1 ~ L5)가 파이프 라인 기법에 따라 연산을 수행할 수 있도록 하더라도, 연산 수행에 필요한 특징값이 이전 콘볼루션 레이어에서 계산되지 않아, 대기해야만 하는 불필요한 대기 시간을 초래하게 된다. 이는 연산기를 효율적으로 사용하지 못하여 발생하는 문제로 볼 수 있다.

[0056] 도 3에서는 최대 연산 시간을 필요로 하는 제2 및 제4 콘볼루션 레이어(L2, L4)를 기준으로 각 콘볼루션 레이어별 연산기 이용 효율과 대기 시간 비율을 함께 표시하였다. 도 3에 따르면, 제2 및 제4 콘볼루션 레이어(L2, L4)는 연산기를 100% 효율로 이용하여 대기 시간이 0클럭인데 반해, 제1, 제3 및 제5 콘볼루션 레이어(L1, L3, L5)는 각각 연산기 이용 효율이 55.6%, 15.6% 및 80%이고, 대기 시간 비율이 44.4%, 84.4% 및 20%임을 알 수 있다.

[0057] 다만 상기한 도 3의 결과는 도 1의 FSRCNN_s에서 다수의 콘볼루션 레이어(L1 ~ L5) 각각이 하나의 연산기를 포함하여 구현되는 것으로 가정하는 경우의 동작으로 각 콘볼루션 레이어(L1 ~ L5)별 연산기의 개수를 서로 상이하게 조절하면, 이러한 비효율성은 크게 개선될 수 있다.

[0058] 따라서 CNN의 하드웨어 구조를 최적화하기 위해서는 각 콘볼루션 레이어(L1 ~ L5)에 포함될 연산기의 적절한 개수를 결정하기 위한 방법이 고려되어야 한다.

[0059] 이에 본 실시예에서는 먼저 다수의 콘볼루션 레이어 중 최종 콘볼루션 레이어가 시스템이 요구하는 처리량의 동작을 수행하기 위해 동시에 수행해야하는 최소 연산 크기를 결정하고, 최종 콘볼루션 레이어에 대해 결정된 최소 연산 크기를 기반으로 이전 나머지 콘볼루션 레이어 각각이 사전에 연산해야 하는 연산 크기를 결정한다. 이는 시스템이 요구하는 처리량을 만족하는 수준에서 연산기의 개수를 최소화해야 하기 때문이다. 그리고 결정된 각 콘볼루션 레이어의 연산 크기에 따른 연산 시간이 가능한 균일해지도록 연산 크기를 조절함으로써, 파이프 라인 기법 적용 시에 대기 시간을 초래하는 병목 현상을 초래하지 않으면서 최소의 연산기를 이용하여 연산이 수행될 수 있도록 한다.

[0060] 도 4는 본 발명의 일 실시예에 따른 CNN의 하드웨어 구조 최적화 방법을 나타내고, 도 5는 도 4의 초기 퓨징 네트워크 설정 단계를 상세하게 나타낸 도면이다.

[0061] 도 4를 살펴보면, 본 실시예에 따른 CNN의 하드웨어 구조 최적화 방법은 우선 CNN에서 최종 콘볼루션 레이어가 시스템 요구 처리량을 만족하도록 출력할 수 있는 최소 크기의 초기 특징값 그룹을 탐색하고, 탐색된 초기 특징값 그룹을 기반으로 이전 특징맵들에서 대응하는 크기의 특징값 그룹을 분석하여 초기 퓨징 네트워크를 설정한다(S10). 최종 콘볼루션 레이어(여기서는 제5 콘볼루션 레이어(L5))는 최종 특징맵이 시스템 요구 처리량을 만족하는 처리량으로 출력될 수 있도록 최소 크기의 특징값 그룹을 설정한다.

[0062] 도 5를 참조하여 특징값 그룹을 설정하는 상세한 단계를 설명하면, 우선 시스템 요구 처리량을 판별한다(S11). 여기서도 상기와 같이, 1920×1080의 해상도를 갖고 60프레임으로 인가되는 FHD 이미지를 3840×2160 해상도를 갖는 UHD 이미지로 업 스케일링하는 CNN인 FSRCNN-s를 가정한다. 이에 시스템 요구 처리량을 만족하기 위해서는 최종 콘볼루션 레이어(L5)가 1920×1080×4의 크기를 갖는 최종 특징맵(FM5)을 초당 60프레임(60 fps)으로 출력할 수 있어야 한다.

[0063] 시스템 요구 처리량이 판별되면, 최종 특징맵의 특징값인 최종 특징값 각각을 획득하기 위해 요구되는 최종 특징값 획득 요구 시간을 판별한다(S12). FSRCNN-s이 UHD 이미지를 60프레임으로 출력하기 위해서는 1920×1080×4의 크기의 프레임 이미지 각각이 16.6(=1,000/60) ms 내에 획득되어야 함을 의미하고, 이는 각 프레임 이미지에 대응하는 최종 특징맵(FM5)의 1920×1080×4개의 최종 특징값 각각이 대략 2.01ns 이내에 획득되어야만 시스템 요구 처리량을 만족할 수 있다는 것을 의미한다.

[0064] 최종 특징값 획득 요구시간이 판별되면, 최종 특징맵에서 지정된 축 방향의 크기에 대한 모든 약수를 추출한다(S13). 최종 특징맵에서 지정된 축 방향의 크기에 대한 약수를 추출하는 것은 파이프 라인 기법 적용 시에 최종 콘볼루션 레이어(L5)에서 수행해야할 연산량을 균등하게 분할하여, 최종 콘볼루션 레이어(L5)에 포함되는 연산기 효율성을 높이기 위해서이다.

[0065] 도 1에 도시된 바와 같이, 최종 특징맵(FM5)의 크기는 1920×1080×4 이고, 여기서는 일 예로 로우(Row) 방향의 크기에 대한 약수를 추출하는 것으로 가정한다. 즉 로우 방향의 크기 1080에 대한 약수의 집합 [1, 2, 3, 4, 5, 6, 8, 9, 10, 12, 15, 18, …, 108, 120, 135, …, 540, 1080]을 추출한다. 그러나 본 실시예는 이에 한정되지 않으며, 칼럼 방향(Cloumn) 또는 깊이 방향(Depth)으로 약수의 집합을 추출할 수도 있다.

[0066] 그리고 추출된 약수의 집합 중 최소 약수(여기서는 1)를 최종 콘볼루션 레이어(L5)를 위한 초기 연산 크기로 설정한다(S14). 이때, 지정되지 않은 축, 즉 칼럼 방향과 깊이 방향의 크기는 지정된 값으로 고정될 수 있다. 일 예로 칼럼 방향에서는 최소값인 1로 고정되고 깊이 방향에서는 최대값인 4로 고정될 수 있다. 따라서 최종 특징맵(FM5)에서 1×1×4 크기의 특징값 그룹이 초기 연산 크기로 설정될 수 있다.

[0067] 초기 연산 크기가 설정되면, 설정된 초기 연산 크기에 따라 최종 콘볼루션 레이어(L5)에 요구되는 연산 시간을 판별한다(S15). 최종 콘볼루션 레이어(L5)의 각 커널이 최소 개수인 1개의 연산기를 포함하도록 구현된다고 가정하면, 각 커널이 1회 대응하는 크기의 연산을 수행하는데 요구되는 시간은 (3×3×32) 클럭(clk)으로서 960ns 이다. 그러나 각 커널이 초기 연산 크기에 대응하는 개수(여기서는 1×1×4)의 연산기를 포함하도록 구현되면, 최종 콘볼루션 레이어(L5)가 최종 특징맵(FM5)의 특징값 각각을 획득하기 위해 요구되는 시간은 (3×3×32)/(1×1×4) 클럭 = (3×3×32)/(1×1×4)×3.333333 = 240ns이다.

[0068] 그리고 판별된 최종 특징값 획득 시간이 시스템의 요구 처리량을 달성하기 위한 최종 특징값 획득 요구 시간을 만족하는지 판별한다(S16). 즉 판별된 최종 특징값 획득 시간이 최종 특징값 획득 요구 시간인 2.01ns 이하인지 판별한다. 만일 최종 특징값 획득 시간이 최종 특징값 획득 요구 시간을 만족하지 못하면, 추출된 약수 집합 중 다음으로 큰 약수를 선택한다(S17).

[0069] 상기의 예에서는 최종 특징값 획득 시간이 240ns 로 최종 특징값 획득 요구 시간인 2.01ns를 크게 초과한다. 그러므로 최종 특징맵을 로우 방향에서 약수 1로 분할하는 경우, 시스템 요구 처리량을 판별할 수 없음을 알 수 있다. 이에 다음으로 큰 약수 2를 선택할 수 있다. 그리고 선택된 약수에 따른 최종 특징값 획득 시간을 다시 판별하고, 판별된 최종 특징값 획득 시간이 최종 특징값 획득 요구 시간인 2.01ns 이하가 될 때까지 반복적으로 다음 크기의 약수를 선택할 수 있다.

[0070] 한편, 판별된 최종 특징값 획득 시간이 시스템의 요구 처리량을 달성하기 위한 최종 특징값 획득 요구 시간을 만족하면, 즉 판별된 최종 특징값 획득 시간이 최종 특징값 획득 요구 시간인 2.01ns 이하이면, 현재 최종 특징맵에 설정된 크기의 특징값 그룹을 시스템 요구 처리량을 만족할 수 있는 최소 크기의 초기 특징값 그룹으로 설정한다(S18).

[0071] 일 예로 다음 크기의 약수를 반복 선택하여 약수 120이 선택된 경우, 최종 특징맵(FM5)에서 1×120×4 크기의 특징값 그룹이 연산 크기로 설정될 수 있다. 그리고 최종 콘볼루션 레이어(L5)의 각 커널이 특징값 그룹의 크기에 대응하는 개수(1×120×4)의 연산기를 포함하도록 구현되는 경우, 최종 콘볼루션 레이어(L5)가 최종 특징맵(FM5)의 특징값 각각을 획득하기 위해 요구되는 시간은 커널 크기/연산기 개수로서 (3×3×32)/(1×120×4)×3.333333333 = 2ns 로 계산된다. 이는 최종 특징값 획득 시간(2ns)이 최종 특징값 획득 요구 시간인 2.01ns 보다 작으므로 시스템의 요구 처리량을 만족한다. 따라서 1×120×4 크기의 특징값 그룹을 최종 콘볼루션 레이어(L5)의 초기 특징값 그룹으로 설정할 수 있다.

[0072] 그리고 설정된 최소 크기의 초기 특징값 그룹을 기반으로 이전 콘볼루션 레이어(L1 ~ L4) 각각이 초기 특징값 그룹에 대응하여 이전 특징맵에서 획득해야하는 특징값 그룹의 크기를 판별하여 퓨징 네트워크를 설정한다(S19).

[0073] 상기한 바와 같이, 최종 특징맵에서 1×120×4 크기의 특징값 그룹이 초기 특징값 그룹으로 설정되면, 이를 기반으로 이전 특징맵에서 대응하는 크기의 특징값 그룹을 판별하여 퓨징 네트워크를 설정할 수 있다. 도 1을 참조하면, 최종 특징맵에서 초기 특징값 그룹의 크기가 1×120×4이면, 퓨징 네트워크을 형성하는 입력 이미지(IN)와 제1 내지 제5 특징맵(F1 ~ F4) 각각에서 특징값 그룹의 크기는 (9×9×1), (5×124×32), (5×124×5), (3×122×5), (3×122×32), (1×120×4)로 설정됨을 알 수 있다.

[0074] 그리고 다수의 콘볼루션 레이어(L1 ~ L5)의 각 커널에 퓨징 네트워크에 따른 특징값 그룹의 크기에 대응하는 개수의 연산기를 할당한다(S20). 즉 제1 내지 제5 콘볼루션 레이어(L1 ~ L5)의 각 커널에는 (5×124×32), (5×124×5), (3×122×5), (3×122×32), (1×120×4)개의 연산기가 할당될 수 있다.

[0075] 이후 할당된 연산기 개수에 따른 각 콘볼루션 레이어별로 요구되는 연산 시간을 분석한다(S30).

[0076] 콘볼루션 레이어(L1 ~ L4)의 각 커널에 설정된 퓨징 네트워크에 따른 특징값 그룹의 크기에 대응하는 개수의 연산기가 할당되면, 각 콘볼루션 레이어(L1 ~ L4)의 각 커널은 할당된 연산기에 따른 연산 시간을 요구하게 된다. 따라서 제1 내지 제4 콘볼루션 레이어(L1 ~ L4)에서 각 커널은 (5×5×1) clk = 83.3ns, (1×1×32) clk = 106.6ns, (3×3×5) clk = 149.9ns, (1×1×5) clk = 16.7ns 의 연산 시간을 요구하게 된다.

[0077] 각 콘볼루션 레이어(L1 ~ L4)에서 요구되는 연산 시간이 분석되면, 분석된 연산 시간 중 최대 연산 시간을 요구하는 콘볼루션 레이어를 판별한다(S40). 여기서는 제3 콘볼루션 레이어(L3)가 149.9 ns 로 최대 연산시간을 요구하는 것을 알 수 있다.

[0078] 또한 최소 연산 시간을 요구하는 콘볼루션 레이어를 판별하고, 판별된 최소 연산 시간 레이어에 대응하는 특징값 그룹을 분할한다(S50).

[0079] 여기서는 제4 콘볼루션 레이어(L4)가 16.7 ns 로 최소 연산시간을 요구한다. 이는 제4 콘볼루션 레이어(L4)가 퓨징 네트워크에 의해 지정된 특징값 그룹을 빠르게 연산하여 획득하더라도, 다른 콘볼루션 네트워크(L1 ~ L3, L5)가 연산을 완료하지 못함을 의미한다. 따라서 제4 콘볼루션 레이어(L4)에 퓨징 네트워크에 따른 특징값 그룹의 크기에 대응하는 개수의 연산기가 할당되더라도 대기 시간이 발생하므로 할당된 연산기가 이용되지 못하는 대기 시간이 발생되며, 이로 인해 연산기 효율성이 저하되게 된다.

[0080] 이에 최소 연산 시간을 요구하는 콘볼루션 레이어가 판별되면, 퓨징 네트워크에서 판별된 콘볼루션 레이어에서 출력되는 특징값 그룹의 크기를 분할하여 수정한다. 이때, 특징값 그룹의 크기는 약수의 개수가 많은 방향으로 분할할 수 있다. 또한 약수 중 1을 제외한 최소 약수에 따라 특징값 그룹의 크기를 분할할 수 있다. 상기의 예에서는 제4 콘볼루션 레이어(L4)에서 출력되는 제4 특징맵(FM4)의 특징값 그룹의 크기가 (3×122×32)이므로, 로우 방향에서의 약수의 개수는 [1, 2, 61, 122]로 4개인 반면, 깊이 방향에서의 약수의 개수는 [1, 2, 4, 8, 16, 32]로 6개이다. 따라서 깊이 방향으로 특징값 그룹의 크기를 2분할하여 (3×122×16)으로 수정할 수 있다.

[0081] 그리고 분할된 특징값 그룹의 크기에 따라 연산기를 재할당하여 연산 시간을 다시 계산한다(S60). 상기의 예에서 제4 특징맵(FM4)의 특징값 그룹의 크기가 2분할되어 (3×122×16)로 줄어들었으므로, 연산기 또한 1/2로 줄여 할당하여 연산 시간을 다시 계산한다. 연산기의 개수가 1/2로 줄었으므로, 연산 시간은 (1×1×5)×2 clk = 33.4ns 로 증가하게 된다. 그리고 계산된 연산 시간이 판별된 최대 연산 시간 이하인지 판별한다(S70). 만일 계산된 연산 시간이 판별된 최대 연산 시간 이하인 것으로 판별되면, 다시 최소 연산 시간 레이어를 판별하고, 판별된 최소 연산 시간 레이어에 대응하는 특징값 그룹을 분할한다(S50).

[0082] 그러나 계산된 연산 시간이 판별된 최대 연산 시간을 초과하면, 현재 설정된 특징값 그룹의 크기를 저장한다(S80). 즉 최대 연산 시간을 초과하지 않는 동안 반복적으로 최소 연산 시간을 갖는 콘볼루션 레이어를 판별하고, 판별된 콘볼루션 레이어에 대응하는 특징값 그룹을 약수의 개수가 많은 방향에 따라 분할하여, 최소 연산 시간을 갖는 콘볼루션 레이어의 연산 시간을 증가시킨다.

[0083] 이와 같이 최소 연산 시간을 갖는 콘볼루션 레이어에 대응하는 특징값 그룹을 반복하여 분할하면, 결과적으로 모든 콘볼루션 레이어(L1 ~ L5)의 연산 시간이 최대 연산 시간에 근접하게 된다. 이때 퓨징 네트워크의 최종 콘볼루션 레이어(L5)의 특징값 그룹의 크기와 최대 연산 시간을 갖는 콘볼루션 레이어에 대응하는 특징값 그룹의 크기는 변화하지 않으므로, CNN의 처리량은 변화하지 않으면서 각 콘볼루션 레이어의 연산 시간이 균등해지므로 대기 시간이 최소화된다. 그리고 상대적으로 연산 시간이 작은 콘볼루션 레이어에 할당되는 연산기의 개수를 반복적으로 분할하여 저감하였으므로, 동일한 처리량을 유지하면서 요구하는 연산기 개수를 줄일 수 있게 된다. 즉 연산기 이용의 효율성을 극대화하게 된다.

[0084] 그러나 이는 최종 콘볼루션 레이어(L5)에 설정된 초기 특징값 그룹의 크기에 따른 초기 퓨징 네트워크에 기반하여 효율성을 극대화한 것으로, 최종 콘볼루션 레이어(L5)의 특징값 그룹의 크기가 변경되면, 즉 퓨징 네트워크가 다르게 설정되면 연산기의 효율성이 더욱 높아 질 수 있다.

[0085] 이에 설정 가능한 모든 퓨징 네트워크에 대해 최대 연산 시간과 최소 연산 시간 사이의 차이가 최소화되었는지 판별한다(S90). 만일 판별되지 않은 퓨징 네트워크, 즉 최종 콘볼루션 레이어(L5)의 지정된 방향의 약수의 집합([1, 2, 3, 4, 5, 6, 8, 9, 10, 12, 15, 18, …, 108, 120, 135, …, 540, 1080])에서 최종 특징값 획득 요구 시간을 만족하는 최소 크기인 120을 갖는 초기 특징값 그룹보다 큰 약수들에 따른 크기를 갖도록 최종 콘볼루션 레이어(L5)의 특징값 그룹을 설정하고, 설정된 최종 콘볼루션 레이어(L5)의 특징값 그룹에 기반하여 이전 콘볼루션 레이어(L1 ~ L4)의 특징값 그룹의 크기를 설정하여 퓨징 네트워크를 재설정한다(S100).

[0086] 그리고 재설정된 퓨징 네트워크를 기반으로 다수의 콘볼루션 레이어(L1 ~ L5)의 각 커널에 퓨징 네트워크에 따른 특징값 그룹의 크기에 대응하는 개수의 연산기를 할당하고(S20), 할당된 연산기에 따른 레이어별 연산 시간 중 최대 연산 시간과 최소 연산 시간을 판별하여, 최소 연산 시간을 갖는 콘볼루션 레이어의 특징값 그룹의 크기를 분할하여 연산기를 재할당함으로써, 콘볼루션 레이어 사이의 지연 시간 차이를 최소화한다.

[0087] 한편 최종 콘볼루션 레이어(L5)의 특징값 그룹의 크기를 가변하면서 설정된 모든 퓨징 네트워크에 대해 각 레이어간 최대 연산 시간과 최소 연산 시간 사이의 차이가 최소화된 것으로 판별되면, 저장된 퓨징 네트워크별 특징값 그룹의 크기를 분석하여, 최소 연산기 개수를 요구하는 퓨징 네트워크의 특징값 그룹의 크기를 판별한다(S110).

[0088] 그리고 판별된 퓨징 네트워크의 특징값 그룹의 크기에 따라 연산기를 할당하여 CNN을 하드웨어적으로 설계한다(S120).

[0089] 결과적으로 본 실시예에 따른 CNN의 하드웨어 구조 최적화 방법은 다수의 콘볼루션 레이어(L1 ~ L5)를 포함하는 CNN에서 파이프 라인 기법에 따라 연산하여 특징맵을 출력하는 다수의 콘볼루션 레이어(L1 ~ L5) 각각이 요구되는 처리량을 만족하는 수준에서, 각 연산 단계에서 가급적 균일한 시간동안 연산을 수행할 수 있도록 함으로써, 지연 시간을 최소화하고 연산기 이용 효율성을 극대화할 수 있다.

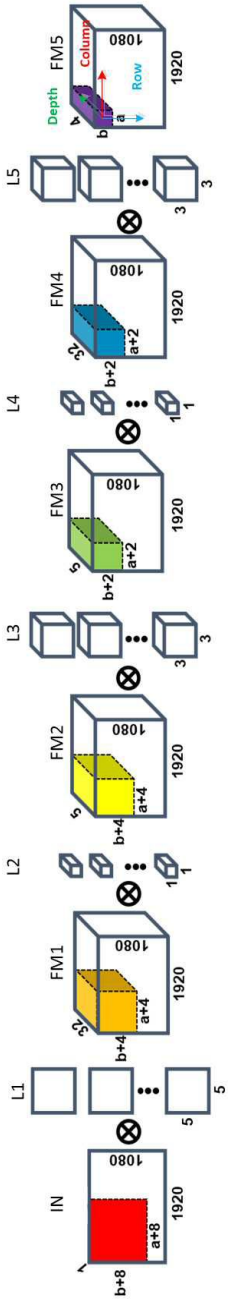
[0090] 상기한 CNN의 하드웨어 구조 최적화 방법은 하드웨어적으로 CNN을 구현하기 위한 하드웨어 설계 장치에서 실행되도록 구현될 수 있으며, 하드웨어 설계 장치는 일 예로 컴퓨터와 같이 지정된 프로그램을 실행하는 연산 장치로 구현될 수 있다. 그리고 본 발명에 따른 방법은 컴퓨터와 같은 하드웨어 설계 장치에서 실행될 수 있도록 매체에 저장된 소프트웨어 프로그램으로 구현될 수 있다. 여기서 컴퓨터 판독가능 매체는 컴퓨터에 의해 액세스 될 수 있는 임의의 가용 매체일 수 있고, 또한 컴퓨터 저장 매체를 모두 포함할 수 있다. 컴퓨터 저장 매체는 컴퓨터 판독가능 명령어, 데이터 구조, 프로그램 모듈 또는 기타 데이터와 같은 정보의 저장을 위한 임의의 방법 또는 기술로 구현된 휘발성 및 비휘발성, 분리형 및 비분리형 매체를 모두 포함하며, ROM(판독 전용 메모리), RAM(랜덤 액세스 메모리), CD(컴팩트 디스크)-ROM, DVD(디지털 비디오 디스크)-ROM, 자기 테이프, 플로피 디스크, 광데이터 저장장치 등을 포함할 수 있다.

[0091] 본 발명은 도면에 도시된 실시예를 참고로 설명되었으나 이는 예시적인 것에 불과하며, 본 기술 분야의 통상의 지식을 가진 자라면 이로부터 다양한 변형 및 균등한 타 실시예가 가능하다는 점을 이해할 것이다.

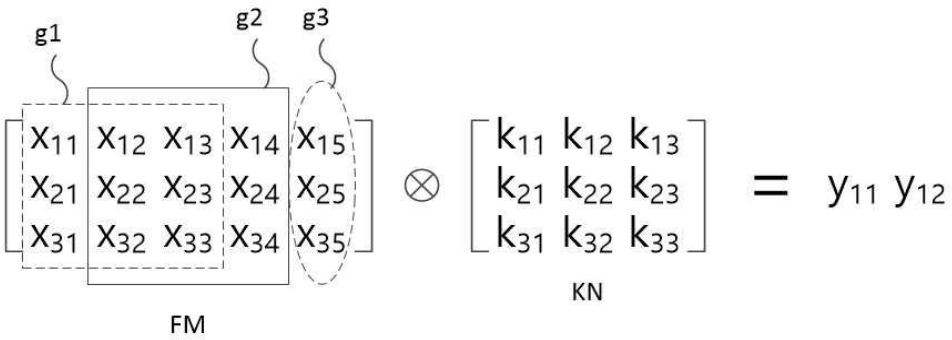
[0092] 따라서, 본 발명의 진정한 기술적 보호 범위는 첨부된 청구범위의 기술적 사상에 의해 정해져야 할 것이다.

도면

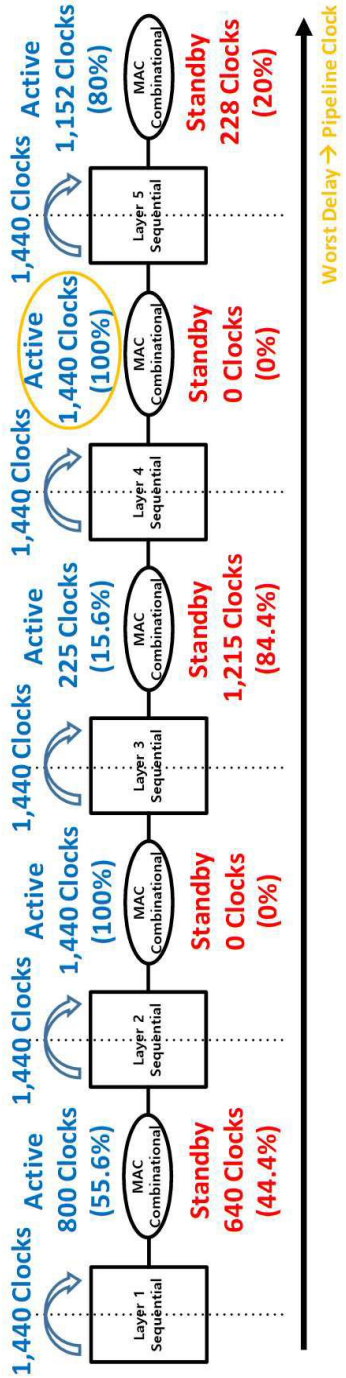
도면1


FM5
Column
Row
Depth
1080
1920
L5
3
FM4
a+2
b+2
L4
1
FM3
L3
FM2
a+4
b+4
L2
FM1
L1
5
IN
a+8
b+8

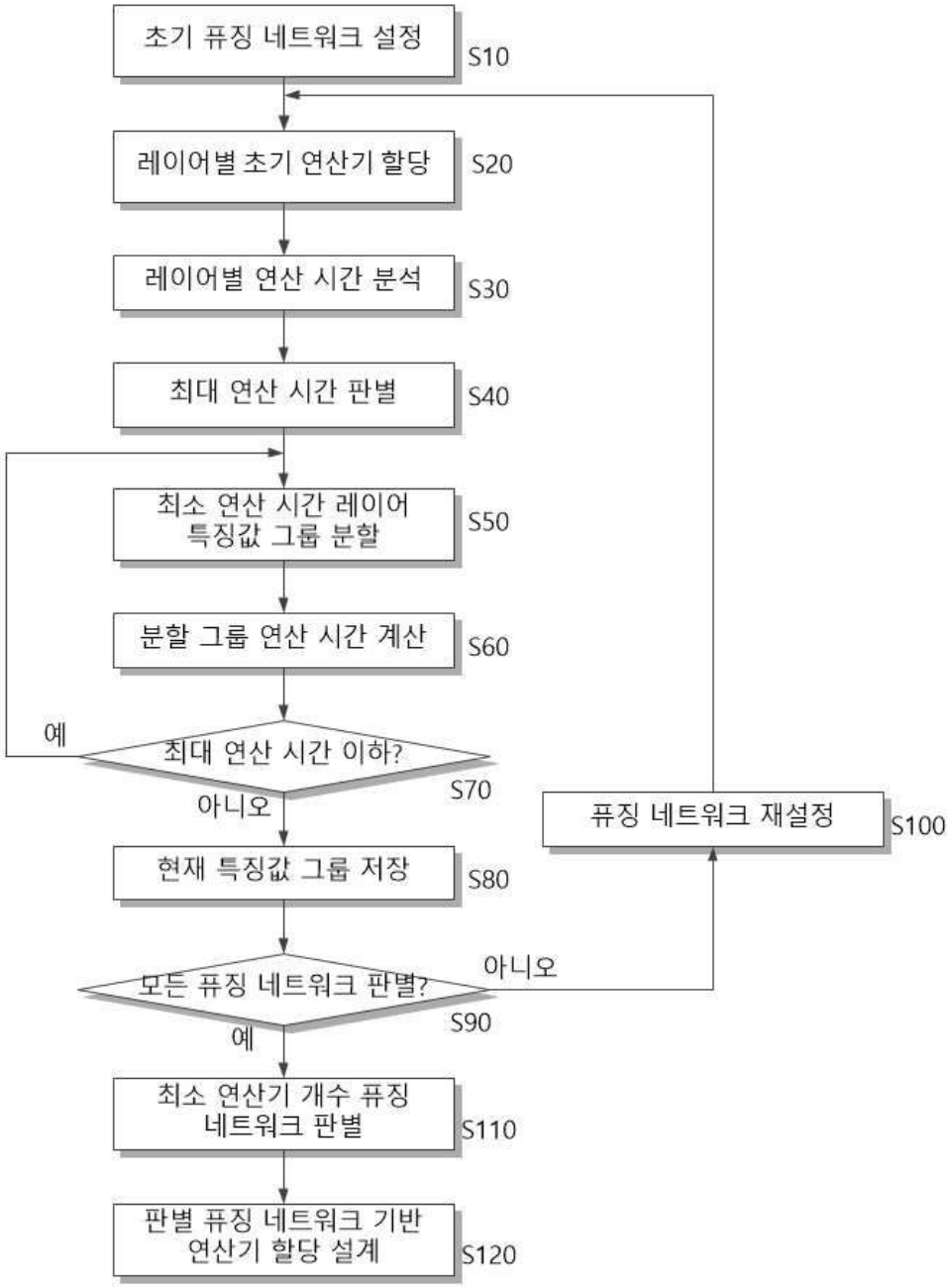
도면2


g1
g2
g3
x11 x12 x13 x14 x15
x21 x22 x23 x24 x25
x31 x32 x33 x34 x35
FM
k11 k12 k13
k21 k22 k23
k31 k32 k33
KN
y11 y12

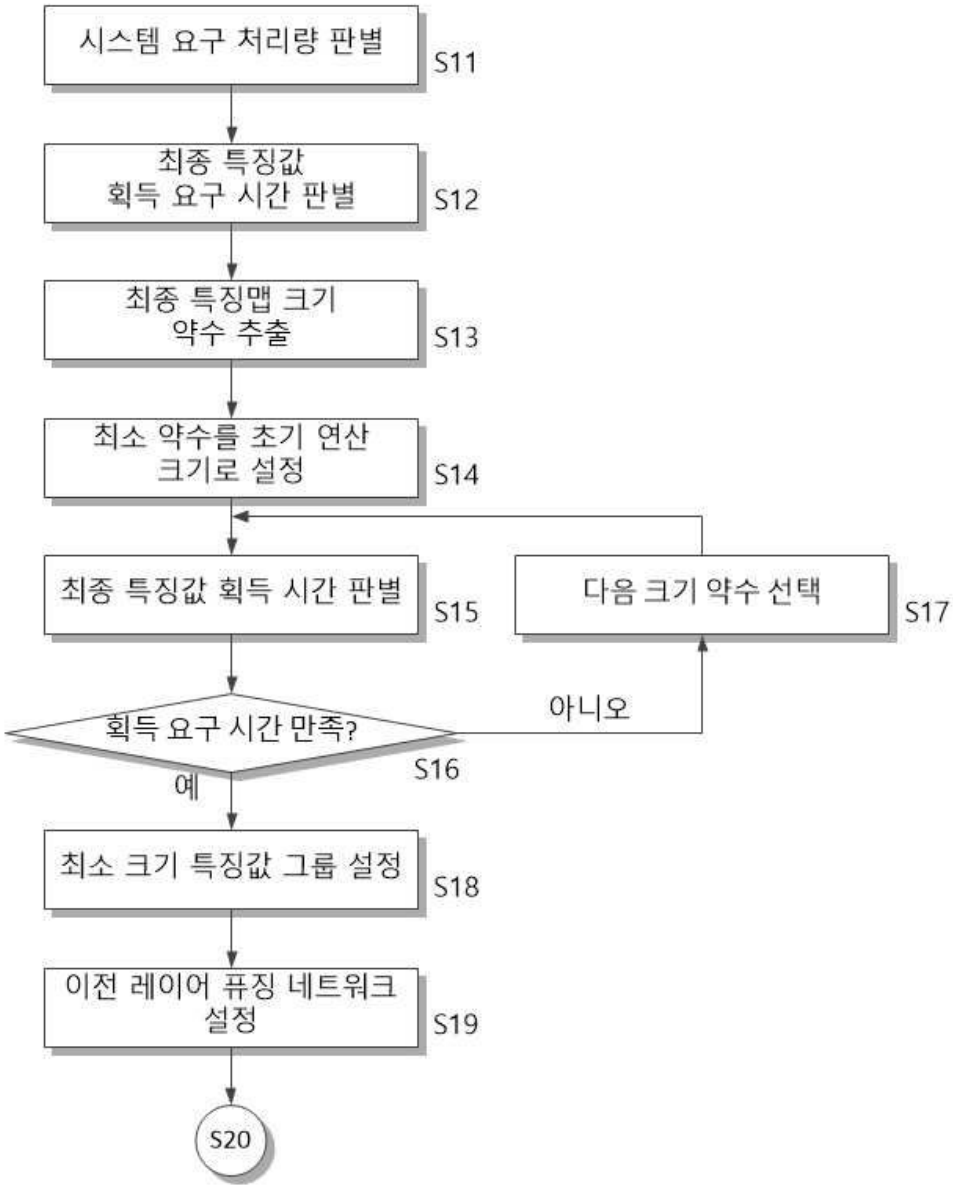
도면3


1,440 Clocks
Layer 1 Sequential
Active 800 Clocks (55.6%)
MAC Combinational
Standby 640 Clocks (44.4%)
1,440 Clocks
Layer 2 Sequential
Active 1,440 Clocks (100%)
MAC Combinational
Standby 0 Clocks (0%)
1,440 Clocks
Layer 3 Sequential
Active 225 Clocks (15.6%)
MAC Combinational
Standby 1,215 Clocks (84.4%)
1,440 Clocks
Layer 4 Sequential
Active 1,440 Clocks (100%)
MAC Combinational
Standby 0 Clocks (0%)
1,440 Clocks
Layer 5 Sequential
Active 1,152 Clocks (80%)
MAC Combinational
Standby 228 Clocks (20%)
Worst Delay → Pipeline Clock

도면4


초기 퓨징 네트워크 설정
S10
레이어별 초기 연산기 할당
S20
레이어별 연산 시간 분석
S30
최대 연산 시간 판별
S40
최소 연산 시간 레이어 특징값 그룹 분할
S50
분할 그룹 연산 시간 계산
S60
예
최대 연산 시간 이하?
S70
아니오
현재 특징값 그룹 저장
S80
모든 퓨징 네트워크 판별?
S90
아니오
예
퓨징 네트워크 재설정
S100
최소 연산기 개수 퓨징 네트워크 판별
S110
판별 퓨징 네트워크 기반 연산기 할당 설계
S120

도면5


시스템 요구 처리량 판별
S11
최종 특징값
획득 요구 시간 판별
S12
최종 특징맵 크기
약수 추출
S13
최소 약수를 초기 연산
크기로 설정
S14
최종 특징값 획득 시간 판별
S15
다음 크기 약수 선택
S17
획득 요구 시간 만족?
S16
아니오
예
최소 크기 특징값 그룹 설정
S18
이전 레이어 퓨징 네트워크
설정
S19
S20